Concept and Implementation of an Adaptive Decentralized Control System for Human-Technology Interaction*

Thomas Kirks[1], Jana Jost [1] and Tim Uhlott[1]

***Abstract*— This paper introduces an architecture for decentralized control systems for human-technology interaction with an elevated focus on adaption and agent mobility. For that purpose, the human worker is represented by a virtual entity a in form of a human agent that allows integration in decentralized control systems. Changes in the population, age structure and the volatility of the employment market demand adequate interface technologies and individualization for the heterogeneous group of employees. The proposed multi-agent system should be open-source and comply with requirements like user integration, interoperability, agent mobility, direct interaction and system adaptivity.**

I. INTRODUCTION

Due to changing requirements in increasingly complex and dynamic industrial environments and growing autonomy level in flexible process chains, multi-agent systems (MAS) have to be extended for the use of human-technology interaction. In the focus of this development, humans have to retain control and also have to be included as part of this system. This architecture aims on the intelligent integration of machines and humans in decentralized control systems. The Internet of Things and Services is a major building block in the fourth industrial revolution which includes methods for simplifying the interaction between machines and humans as a result of adaptive systems. Especially, warehouse logistics and production facilities the developments of Industry 4.0 face changes in the population and employment market, besides rising complexity. People work as leased employees, thus not being able to fit right into the process from the beginning or being of higher age with the necessity to earn one's living, face impairments and need support more often. In relation to Industry 4.0, it is straight-forward to represent the human worker by a virtual entity and let this entity support the worker by mitigating problems in place of him or her and adapt to specific situations. It makes sense to deploy these entities with human interface devices like wearables. Wearables and human interface devices available on the market today, offer many features, are quite powerful and appear in different types of variation, so that they are very useful interfaces for human-technology interaction (HTI). There are smartphones, smartglasses and smartwatches, only to mention the most common. These devices partially offer high performance CPUs and relatively high storage capacities. The operating systems like Android or iOS offer access to network communication and inbuilt sensors. Not only display, microphone and audio speakers feature many communication channels for the user and his interaction in technology-driven production and logistics systems. Unfortunately, state-of-the-art frameworks for multi-agent systems do not provide functionality to implement a lightweight adaptive decentralized control system for human-technology interaction with the features of interoperability, agent mobility and real decentralized multi-agent design at the same time. The following work proposes an architecture which bypasses these disadvantages.

*This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688117 (SafeLog).

[1]Thomas Kirks, Jana Jost and Tim Uhlott are with the Department Automation and Embedded Systems, Fraunhofer Institute for Materialflow and Logistics, 44227 Dortmund, Germany thomas.kirks@iml.fraunhofer.de jana.jost@iml.fraunhofer.de tim.uhlott@iml.fraunhofer.de

II. BACKGROUND

A. Changes in the Population and Employment Market

The demographic change will fundamentally alter the population structure of most industrialized nations in the next couple of decades. On the one hand, the life expectancy of people born in 2010 to 2015 will rise about 8.45% in 2045-2050 globally. On the other hand, although the fertility levels are still varying a lot across countries and regions the level of total fertility has already been reduced and is still declining. While in 1975 to 1980 23% of women gave birth to five children and only 21% of all women had only less than 2.1 births, in 2045 to 2050 high fertility rates with more than five births per woman will not exist any longer. Instead, 69% of all women will have less than 2.1 children. Rising life expectancy paired with declining fertility leads to population ageing. While in 2017 already 13% of the global population is aged 60 or over, in 2050 people aged 60 or more will constitute a quarter or more of the population of all world regions except Africa [1].

Besides an aging population, job markets in many countries have to deal with skilled worker shortages in different fields [2]. Since 2018 the number of job openings in the United States is higher than unemployed persons [3]. Further, international migration to Northern America, Oceania and Europe has been and will be a key feature of global migration patterns [1].

All these facts lead to new challenges for companies. Older as well as migrated people have other demands than younger and national citizens. The employment market as well as the job profiles have to become more flexible. The on-boarding process for new employees should be shortened so that employees are flexible and universally deployable. Therefore, the way of information transmission has to be

978-1-7281-5411-4/19/$31.00 ©2019 IEEE
DOI 10.1109/SOCA.2019.00026

changed dramatically. Not only different languages have to be considered. Further, a variety of cultures, different working experiences and skill levels have to be regarded.

B. Social Networked Industry

The convergence of physical and virtual worlds (known as Industry 4.0) is rapidly increasing throughout all kinds of businesses. Associated with this, the usage of different technologies and processes as well as divergent interests a common understanding is required. For achieving this, the reference architecture model for Industry 4.0 (RAMI4.0) was introduced (see Fig. 1).

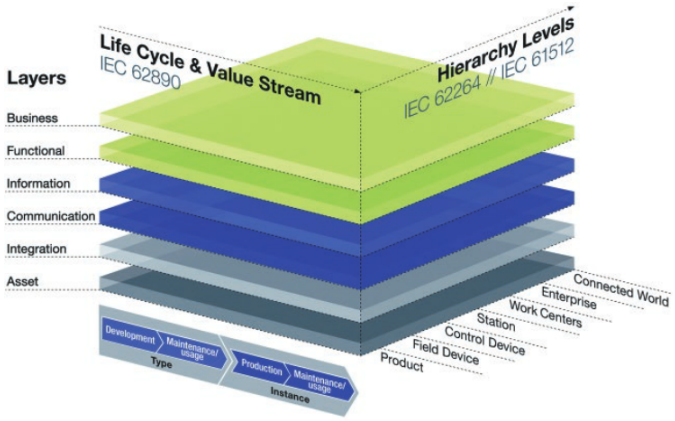

Fig. 1. Reference architecture model for Industry 4.0 (RAMI4.0) [7]

The reference architecture represents the Industry 4.0 space in the three dimensions: layers, life cycle and value stream and hierarchy levels. The layer dimension deals with the various perspectives of a component, how they interfere with one another and includes all its functionality. To display the relationships as well as links the second axis is used and is based on the draft of IEC 62890. The right-hand horizontal axis follows the IEC 62264 and IEC 61512 standards and represents the location of functionalities and responsibilities within the factories and plants[7]. Although the human being is mentioned shortly as part of the Asset Layer in the RAMI4.0, further thoughts for integrating him or her better into the processes of the Industry 4.0 are needed. In the context of RAMI4.0, the human being will be designed as the administrative shell depicted in Fig. 2.

Industry 4.0 is a socio-technical system in which humans and cyber-physical systems are working hand in hand. New forms of cooperation between humans and machines arise through the way communication takes place. This is referred to as "Social Networked Industry" (SNI). Not only are components connected on a vertical level also the connection over company boundaries (horizontal connection) plays a major part. [6] Through the Internet of Things different types of devices can exchange a variety of information. In the consumer and private field the usage of social networks is a daily routine. While bringing both views together, the SNI ensures a comprehensive approach for all three components (humans, organization and technology) of socio-technical

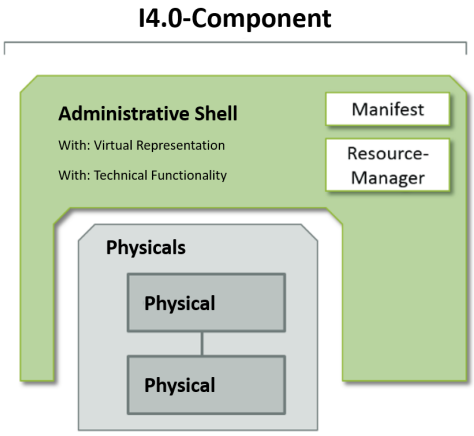

Fig. 2. Administrative Shell of an I4.0-Component cf. [7]

systems and offers new possibilities for human-technology interaction[4].

C. Decentralized Control Systems

Agents in computer science are defined as programs, which can fulfil tasks autonomously. They are described by features and behaviours. They interact with each other using certain communication infrastructures such as LAN or WLAN. Software-agents are defined by their behaviour. In general, there are reactive agents, adaptive agents and cognitive agents. Additionally, their autonomy is described by features like persistence, social ability, robustness, communication ability, and reactivity [5]. A system of agents is called multi-agent system (MAS). Due to its autonomous behaviour MAS can be used for decentralized control systems.

JADE [10] is well known for years and implements FIPA Agent Communication Language Specifications and provides agent mobility but needs a main container to connect to, which portrays a single point of failure (SPOF). ROS is very powerful and provides a communication framework and a great community, but is limited to Linux and only partially to Windows operating systems and needs an instance called roscore, which constitutes also a SPOF [11]. ROS 2 is about to change these drawbacks and will be available on Windows and macOS as well [12] - but it is not able to run on Android, iOS as of now, although there are attemps to fix this [13]. For a detailed look into existing MAS frameworks and comparison, Iñigo-Blasco [14] collected an extensive compilation on these frameworks.

III. MOBILE HUMAN AGENT

The proposed system is based on the concept that there is a digital representation of the human worker - the human agent. The human agent should run on wearable interface devices but is not limited to. On the one hand, it enables the worker to be part of a decentralized control system, to communicate directly with machines and robots and interact with them. On the other hand, the machines are aware of the human and are able to react and most conveniently adapt to the users behavior and needs.

A. Requirements

The conception of a virtual representation of the human worker, the human agent, is subject to a variety of requirements. Besides the requirements, which focus on the way of representing the worker with all his properties and abilities, the following seven fundamental demands have to be matched to ensure an effective integration of the human into processes of the SNI:

- Identification,
- Localization,
- Integration,
- Interoperability,
- Mobility,
- Interaction and
- Adaptivity.

The first five requirements mainly are requirements of the actual multi-agent system to achieve effective integration of the human and realizing the mobility of the human agent when changing interface devices like wearables. The last two requirements have to be met when realizing cooperation between humans and robots, make them aware of each other and make technology adapt to the human's needs.

To operate in a given decentralized environment with other entities the human agent himself has to be identified and has to be able to identify other entities. Further, to enable him to work together with other agents he needs to know and sustain his pose in an environment (localization). Depending on the device the human is using, the human agent can localize himself or can obtain a position via other agents close by proposed by Kirks et al. [9]. By being able to identify and localize oneself and other entities and by depicting the services the human worker is offering, the integration into the overall multi-agent system can be realized. Therefore, the human worker is represented as a software agent which can run on different interface devices. The human agent offers the same functionalities as any other agent in the multi-agent system. Further, the human agent has to function as interface to different machines and technologies. It has to transfer the offers and needs of the worker into the correct message format used by existing communication techniques of the multi-agent system.

Since different roles e.g. picker or service technician, demand different devices during the process and due to the volatile employment market, workers might be appointed flexibly and might change roles during the day - interoperability of the human agent has to be granted. The agent should be running on any kind of device e.g. smartphones, smartwatches, wristbands or smart glasses - respectively on any operation system e.g. iOS or Android.

Mobility might be, according to nowadays flexible job markets, the most important one to ensure an effective system. The human agent needs to be mobile. This means, that the agent should be moved from one human interface device to another if the worker changes his role and therefore the device. Further, once moved, the human agent has to adapt directly to the device features e.g. using different ways of presenting information. Due to the "Social Networked Industry" a horizontal connection over company boundaries takes place. The human agent has to be mobile in a way that the worker can use him wherever he has to work - on the shop floor or in another company.

To counteract any adverse effects of the changes in the population and employment market (see II-A) the human agent has to adapt to the represented human. This means it has to cope with the individual needs of the represented worker. For example, during the on-boarding process the worker might need more assisting information for a given task, which can be omitted once he has gained more knowledge about it. Also, the way a worker wants to interact with technology demands adaptation. In the beginning, for instance, the worker might not be used to work with robots and feels uncomfortable or afraid of them. Proxemic distances, which the robot should keep between the human and itself, can be provided by the human agent and the robot can adapt to them.

B. Architecture

We propose a flexible, modular architecture (Fig. 3) and an implementation which complies to the requirements mentioned before, reflects RAMI4.0 and the administrative shell of an Industry 4.0 Component and uses preferably open source libraries for mitigation of implementation barriers. In the Administrative Shell the Physical represents the human and the human agent complies to the Virtual Representation and Technical Functionality demands as well as the Manifest (by the use of the Configuration Component) and the Resource-Manager (State Component cf. Fig. 2).

In the sense of service orientation the architecture shall allow for easy service announcement and service discovery in a way that new participants can share their information and existing participants can find those as well.

The agents can run on the device itself, whether mobile device for the human agent or the controller board of robots. They can also be run on dedicated platforms providing enough performance and data storage or to fit the existing software structures in a company. This also might be the case, if the agent communication using the proposed media can not be provided on the physical itself. Furthermore, it is thinkable to have the agent run as cloud service and communicate via the hardware abstraction layer with the actual physical.

For the identification of the user, the hardware, the agent, the agent container and messages we use universally unique identifiers (UUID; GUID in context of .NET) which will be randomly generated and assigned for each required architecture component. Thereby, we gain control and traceability and enable the mobility of human agents.

The components use method calls and events to update or change states. This enables the architecture to be modified and exchanged with different implementations of the component if necessary - in that way we gain software modularity.

The following paragraphs depict the nature and functions of the proposed components of the architecture.

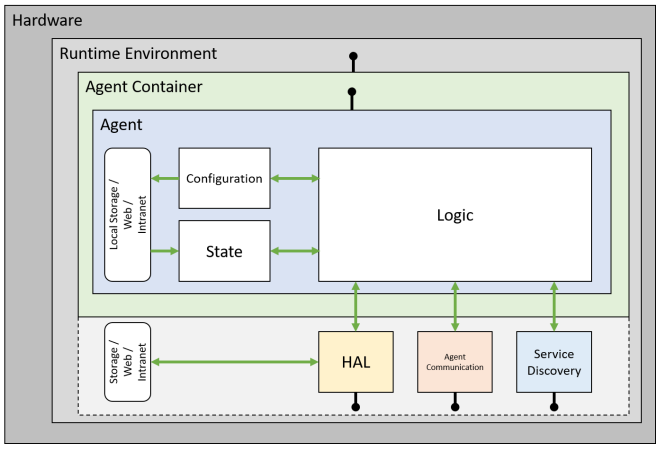

Fig. 3. Proposed Architecture

1) Hardware and Runtime Environment: For interoperability and integration requirements, we want the proposed control system to run on (almost) any hardware that is able to run operating systems like Linux, Windows or macOS. The hardware can be production machines, robots or other things or devices that are even not capable to run the agent container and agent. In that case, there should be a dedicated hardware capable of doing so. To influence the state of the digital representation of this physical entity, we depend on sensor systems or other agents to determine the actual state of this entity.

2) Hardware Abstraction Layer: In a conventional sense, for electronic devices the hardware abstraction layer provides access to sensors, actuators and information of the hardware for the application layer. In the sense of the proposed architecture, the HAL allows to access sensors, actuators and functionality of robots and use it in the application logic of the agent. As an example, the data of the laser rangefinder of an autonomous transport vehicle (ATV) can be accessed and forwarded through the logic via the agent communication using the publisher pattern. For each physical, respectively the robots and machines, in the multi-agent system, the HAL provides this data to all dedicated agents. Initially, it can be configured from storage, web or local network by reading configuration files. For the human to be part of the MAS, the HAL encapsulates individual properties and abilities in the human agent allowing it to provide and access adequate services in the system. Exemplarily, the HAL for a human agent encapsulates properties like physical features (e.g. height of a person), proxemic information (e.g. comfortable distances towards ATV) or impairments (e.g. red-green color blindness). With this information provided to the MAS, robots can adapt to the individual and adjust the height of the load handling device (LHD) to an ergonomic height or the presentation of information of the used wearable can be adapted to the red-green color blindness and show a different color theme in the application. Abilities, like being able to transport small load carriers (SLC) manually, can be defined making it possible for the human agent to provide the service of SLC transport on the shop floor. Also restrictions due to approvals in the business environment (e.g. the company approves a person to gain access to the inbound process tasks in a warehouse) can be mapped. Finally, the HAL gives access to existing multi-agent systems used by the physicals consequently connecting the proposed architecture to other MAS of the represented physical.

3) Agent Communication: The communication between the agents in the proposed architecture is basically built upon the publish-subscribe pattern - but it can also be extended to use other communication schemes. All information that needs to be exchanged between agents is realized in this manner. We designed CommunicationObjects that contain a header with GUID, timestamp, topic name, source information and a ContentObject for each message. The implementation of the ContentObject depends on the intended use for services and related data it only contains a GUID. Implementations of the ContentObject, for example for debug purposes (a DebugContentObject), may contain additional data like the debug message and further annotations.

4) Service Discovery: This component manages all services provided by an agent. It looks up available services of all agents in the network and announces agent-specific services. These services depict endpoints for communication and information interchange in the MAS. Between endpoints agents communicate using the publish-subscribe pattern [8]. The retrieved data is accepted by the logic component and affects the agent state and behaviors. At runtime, these endpoints can be connected making the architecture fully decentralized.

5) Agent Container: The Agent Container encloses a single agent - either human agent or any other agent. The contained agent can be loaded or offloaded dynamically at runtime. When the State and Configuration of an agent is saved and transferred to another Agent Container, it is possible to migrate an agent from one runtime environment to another, hence satisfying the requirement for mobility.

6) Agent: The Agent itself contains the components Logic, State, Configuration and Storage. This is the basic representation of the physical or human individual. The logic is connected to the HAL, Agent communication and the service discovery.

7) Logic: In the Logic component, behaviors are implemented. Different realization of bahavior scheduling are possible here. The Logic controls the internal agent state and updates it via information from the HAL, Agent Communication or internal procedures. The logic also configures the Service Discovery depending on the

Configuration component and behaviors or properties.

8) State and Configuration: On start-up of an agent, the configuration and the last known state of the physical (properties of the user) can be loaded from a local storage, a web interface or the network. On shut-down, the configuration and state are backed up again. These components are also used when an agent changes execution environment from one user interface device to another. In this case the state and configuration is saved and then transferred to the intended Agent Container on another interface device.

C. Test Implementation

The test set-up consists of two different robots, one runs a PLC and provides a proprietary network protocol for control and state retrieval, the other one is a ROS-based robot with Ubuntu 16.04 LTS on an NVIDIA Jetson TX2. Furthermore, we used a Windows Laptop, a Raspberry PI 3, a Microsoft HoloLens and a Lenovo Phab 3 (Android). The ROS robot is able to adjust its load handling device in height and the second robot can be navigated manually using the control protocol. The Android phablet is used to control the second robot via the human agent. The HoloLens uses the localization and its tracking feature to calculate a relative distance to the ROS robot (following Kirks et al. [9]) and also provides an individual height for the LHD. The Raspberry Pi simply retrieves the actual pose of the human agent running on the HoloLens. The Windows Laptop is the development device and the default test environment for the implemented agents.

For the interoperability requirement, we have implemented the proposed architecture in .NET and compiled a dynamic link library containing the main components of the agent system. For specific deployments (Android vs. Windows) we have set up different Visual Studio projects and referenced the library. We used Xamarin for the implementation of the human agent on the Android device. For execution on Linux devices, we simply needed to install Mono and run the executable. The agent communication is realized using NetMQ publishers and subscribers (a .NET implementation of ZeroMQ [15]). We use the CommunicationObjects and derived ContentObjects for specific message payloads (cf. section III-B.3) and serializing them with Google Protocol Buffers [16].

The requirement for integration is realized by the use of wearables for human workers and the implementation of human agents of the proposed architecture on the wearables themselves. Furthermore, the same architecture is used for machine dedicated agents to provide agent communication and information interchange between machines and humans. Meaning we used the same MAS on the robots and implemented the HAL for them according to the use case. Furthermore, we implemented Zeroconf for service discovery [17]. In that way agents can announce their services in the network and can discovery others at runtime. This allows for dynamic integration of agents in the system and complies with a decentralized structure.

The tests on the different devices and platforms turned out to work as expected. Communication between robot agents and human agent was applicable and the state retrieval and control of the robot with the use of human agents was possible, too. The successful functionality test for adaption of the ROS robot to the individual proxemic distance and height adjustment of the LHD makes us consider this MAS architecture as an appropriate system for human-technology interaction.

IV. CONCLUSION

This paper shows an architecture and test implementation for an adaptive decentralized control system for human-technology interaction and takes one step closer towards the “Social Networked Industry”. The human software agent was designed with seven fundamental characteristics which ensure adaptivity in human-technology interaction for heterogeneous multi-agent systems. With this architecture it is possible to connect users to existing MAS and allow direct interaction. The integration of the concept into existing Industry 4.0 systems is feasible by the extension of the broadly used Reference Architecture Model for Industry 4.0.

The aging population, international migration as well as the lack of skilled employees lead to the demand of easy ways to flexibly integrate the human worker in complex systems and to adjust to his individual needs. The human agent with its adaptivity and mobility offers a broad usage without less limitations due to numerous operation systems, interface devices or company boundaries.

V. OUTLOOK

As of now, the actual test implementation represents the proposed architecture but it is a quite static approach and does not support dependency injection. We will change that in the following development. As we have tested the implementation on Windows, Debian, Android and Windows Mixed Reality for the HoloLens, we will test it on iOS and macOS, too. Later we run tests on the different platforms to find out about performance and data throughput. For security and performance reasons, we want to evaluate different methods to store personal data. The user should retain control over his personal data, so the question arised: where to store the human agent’s state or configuration, which is actually part of personal data? We want to consider the use of conventional databases or blockchain or smart contracts technologies.

Additionally, we want to evaluate the usability of the developed heterogeneous multi-agent systems. Therefore, the relocation of the human agent on different devices and throughout a variety of processes will be tested with subjects in industry-oriented use cases.

Finally, we intend to provide the test implementation to the community and make it available open-source. Examples we implemented already and more to be developed to prove realization of the mentioned requirements, will be published.

REFERENCES

[1] United Nations, Department of Economic and Social Affairs, "Population Division, World Population Prospects: The 2017 Revision", Volume II: Demographic Profiles (ST/ESA/SER.A/400), 2017.

[2] Europäisches Zentrum für die Förderung der Berufsbildung, "Fachkräftemangel und -überschuss in Europa", November 2016.

[3] Bureau of Labor Statistics, "Job Openings and Labor Turnover Survey Highlights", 2019.

[4] C. Prasse, C. Tüllmann, A. Nettsträter, and M. t. Hompel, "Social Networked Industry - ganzheitlich gestalten", in Future Challenges in Logistics and Supply Chain Management, 2018.

[5] M. Wooldridge and N. R. Jennings, Intelligent Agents: Theory and Practice in Knowledge Engineering Review, 1995.

[6] M. t. Hompel, M. Putz, and A. Nettsträter, "Social Networked Industry" (Whitepaper), 2019.

[7] ZVEI and VDI, Reference Architecture Model Industrie 4.0 (RAMI4.0) (Status Report), July 2015.

[8] K. Birman and T. Joseph. 1987. Exploiting virtual synchrony in distributed systems. In Proceedings of the eleventh ACM Symposium on Operating systems principles (SOSP '87). ACM, New York, NY, USA, 123-138.

[9] T. Kirks, J. Jost, T. Uhlott, and M. Jakobs, "Towards Complex Adaptive Control Systems in Intralogistics", in The 21st IEEE International Conference on Intelligent Transportation Systems, IEEE, 2018.

[10] Jade Site, "JAVA Agent DEvelopment Framework", 2019. [Online]. Available: https://jade.tilab.com/. [Accessed: 20-Apr-2019].

[11] Open Source Robotics Foundation, Inc., "ROS - The Robot Operating System", 2019. [Online]. Available: http://www.ros.org/. [Accessed: 20-Apr-2019].

[12] Open Source Robotics Foundation, Inc., "Changes between ROS 1 and ROS 2", 2019. [Online]. Available: https://design.ros2.org/articles/changes.html. [Accessed: 20-Apr-2019].

[13] Esteve Fernandez, 2018, "ROS2 for Android, iOS and Universal Windows Platform", https://roscon.ros.org/2018/presentations/ROSCon2018_ROS2 for Android, iOS and Universal Windows Platform.pdf. [Accessed: 20-Apr-2019].

[14] P. Iñigo-Blasco, F. Diaz-del-Rio, M. C. Romero-Ternero, D. Cagigas-Muñiz, and S. Vicente-Diaz. 2012. "Robotics software frameworks for multi-agent robotic systems development." Robot. Auton. Syst. 60, 6 (June 2012), 803-821.

[15] NetMQ, 2019, "NetMQ", https://netmq.readthedocs.io/en/latest/. [Accessed: 20-Apr-2019].

[16] Google Developers, 2019, "Protocol Buffers", https://developers.google.com/protocol-buffers/. [Accessed: 20-Apr-2019].

[17] S. Cheshire, B. Aboba, and E. Guttman, "RFC 3927 - Dynamic Configuration of IPv4 Link-Local Addresses", IETF, 2005.